AMENDMENT TO THE SPECIFICATION

Please amend the specification by marked up replacement paragraph(s) as follows.

On page 2, please replace paragraphs 1 and 2 as follows:

[01] The present application is related to U.S. Patent Application Serial No. [[_____]] 10/140,818 entitled "Method and Apparatus for Change Data Capture in a Database System" filed on [[_____]] May 9, 2002 ~~(attorney docket no. 50277-1002, client docket no. 2000-190-01)~~ by William D. Norcott et al~~., the contents of which are hereby incorporated by reference~~.

[02] The present application is related to U.S. Patent Application Serial No. [[_____]] 09/863,491 entitled "Asynchronous Change Capture for Data Warehousing" filed on [[_____]] ~~(attorney docket no. 50277-1005, client docket no. 2001-003-01)~~ May 24, 2001 by William D. Norcott, the contents of which are hereby incorporated by reference.

On page 4, please replace paragraph 8 as follows:

[8] Conventional systems have used triggers for synchronous change data capture, either by using the CREATE TRIGGER statement or by using an internal mechanism with equivalent functionality. A trigger is an object that specifies a series of actions to be automatically performed when a specific event occurs, and, according to industry standards, the events that cause triggers to be activated (or "fired") are DML statements. For synchronous change data capture, triggers have been designed to fire when a row of a database table is updated, inserted, or deleted. Each of these modifications is associated with its own system change number (SCN), which is recorded by the trigger. The true commit system change number (CSCN) for the transaction cannot be recorded at this time because the commit has not yet occurred. Thus, existing trigger-based mechanisms do not capture the CSCN, which is a serious technical flaw that greatly limits their usefulness. Without the CSCN for each record, it is not possible to identify changes that occurred within a particular transaction nor to consider the effects of each transaction in turn, in the precise order in which the changes ~~where~~ were committed to the relational database management system.

On page 6, please replace paragraph 14 as follows:

[14] In one implementation, the system change number recorded during the commit is generated after ~~obtained~~ obtaining a concurrency lock, which ~~prevent~~ prevents other processes from generating new system change numbers. After recording the transaction identifier and the system change number in the second database object, the concurrency lock is released.

On pages 10-11, please replace paragraph 29 as follows:

[29] An asynchronous extraction mechanism may also be employed, in which a log shipper 119 periodically copies recovery logs 117 that are produced by the OLTP database 113 in the normal course of operation. The recovery logs 117 contain all the changes that have been applied to the OLTP database 113 and are used for backing up the data in the OLTP database 113 and restoring the data in case of a system crash. The log shipper 119 copies the recovery logs 117 to an area of the staging system 120 called a change source 131, which can be implemented as an operating system directory. The publisher 127 interacts with a log viewer process 129 to obtain the change data from the shipped recovery logs in the change source 129 without having to be aware of the internal implementation details of the recovery logs. The publisher 127 then loads the change data obtained via the log viewer process 129 into the change tables in the analysis database 125. The asynchronous extraction mechanism is described in greater detail in the commonly assigned, co-pending U.S. Patent Application Serial No. [[_____]] 09/863,491 entitled "Asynchronous Change Capture for Data Warehousing" filed on [[_____]]May 24, 2001 ~~(attorney docket no. 50277-1005, client docket no. 2001-003-01)~~ by William D. Norcott, the contents of which are hereby incorporated by reference.

On pages 11-12, please replace paragraphs 30-32 as follows:

[30] In accordance with one aspect of the present invention, the change data extracted from the OLTP database 113 is maintained in one or more database objects, referred to herein as "change tables" and "transaction tables" under control of a database management system, e.g. analysis database 123. Referring to FIG. 2 by way of example, each source table or database object on the OLTP database 113 that is subject to change data capture is associated with a corresponding

change table 211, 213, 221, 223 in the analysis database ~~123~~ 125. For transactional consistency, change tables 211, 213, 221, 223 are grouped into sets of one or more "change sets " 210, 220 such that the publisher ~~125~~ 127 ensures that all new change data added to the change tables in the same change set (e.g. ~~changes~~ change tables 211, 213 of change set 210) are added at the same time, e.g. the modifications to these ~~changes~~ change tables are performed in the same transaction and committed. In the example depicted in FIG. 2, there are two change sets, change set 210 and change set 220. Change set 210 comprises change table 211 and change table 213, which correspond to respective tables (not shown) on the OLTP database 113. Likewise, change set 220 comprises change table 221 and change table 223, which also correspond to respective tables (not shown) on the OLTP database 113. The information that defines the structure of the change sets 210, 220 and change tables 211, 213, 221, 223 is maintained in system metadata 230.

[31] Each change table employs a number of control columns in addition to the source table columns whose values were extracted, transported, and loaded from columns of the corresponding source table in the OLTP database 113. In the example of FIG. 2, change table 223 is depicted as having a set of source table columns 231 and control columns XID 233, TIME 235, OP 237, and ROW 239. The source table columns 231 may include all or a subset of the columns in the corresponding source table. In various implementations, the control columns may be part of the same database object that contains the source table columns or part of a parallel, associated database object, which can be joined with source table columns (e.g. by a row identifier or a primary key).

[32] The control columns XID 233, TIME 235, OP 237, and ROW 239 preferably have reserved names that customers are not allowed to use for their own columns, for example, names with a reserved character such as a dollar sign ($). The reserved names, of course, can be any valid string and, in this example, are named XID 233, TIME 235, OP 237, and ROW 239 for mnemonic reasons. The XID 233 column holds a transaction identifier, which can be a monotonically increasing number that uniquely identifies each transaction on the OLTP database 113 that gave rise to the change data.

On page 12, please replace paragraph 34 as follows:

[34] The TIME 235 column contains the commit time of the transaction that gave rise to the change data. This column helps subscriber applications 121 select or view change data that occurs in particular periods of time.

On pages 13-14, please replace paragraphs 37-38 as follows:

[37] Although not depicted in FIG. 2, additional control columns may be provided to facilitate the implementation of embodiments of the present invention. For example, a bit mask of the updated columns can be used to identify quickly which columns have changed. As another example, the name of a user who causes the operation can be recorded in a control column. The row identifier of the affected row in the source table can also be included in a control column.

[38] In addition, a separate transaction table ~~240~~ 250 is provided to record the transaction identifiers 241 of committed transactions and the system change number 243 that is associated with the transaction commit. A session identifier 245 may also be provided to identify the current database session, differentiating among all current users of the ~~relation~~ relational database management system. The transaction table ~~240~~ 250 may be configured to be a system-wide table that is applicable to all users.

On page 14, please replace paragraphs 39-40 as follows:

[39] FIG. 3 is a flowchart that illustrates the operation of one embodiment of the present invention. At step 301, a user begins a transaction in which one or more operations (e.g. SQL or DML statements) are to be performed. At step 303, a unique transaction identifier is generated to identify the operations in the change table ~~233~~ 223 that belong to the same transaction.

[40] For each operation in the transaction (controlled at step 305), the relative sequence identifier is generated (step 307) and recorded in the change table ~~233~~ 223 along with the transaction identifier, an indicator of the kind of operation (e.g. update, insert, delete), the change data for the operation. The change data may comprise, for example, the values of all columns that have changed in the source table, including, in the case of an update operation, both the new and old values of any columns that were updated. In this embodiment, the system change number for the operation need not be directed in the change table itself.

On page 15, please replace paragraph 44 as follows:

[44] At step 313, a system change number for the current statement is allocated for use as an approximate commit system change number ACSCN. The approximate commit system change number is less than the true commit SCN that is about to be generated later in the commit–at a point where it is too late to make use of it. Because the approximate commit system change number (ACSCN) is obtained under a concurrency lock, no change data capture operation in any other transaction is permitted to ~~allocated~~ allocate a system change number as long as the concurrency lock is being held. Thus, the approximate commit system change number constitutes a reliable proxy for the true commit SCN.

On pages 15-16, please replace paragraphs 47-49 as follows:

[47] Accordingly, the transaction table ~~240~~ 250 maintains a row for each transaction that was captured synchronously (identified by the transaction identifier 241), and the transaction table maintains records of the true order in which all transactions committed (as determined by the approximate commit system change number 243). In terms of performance, it has been found that the worst-case overhead of capturing the commit system change number in a transaction table ~~240~~ 250 is about 3/4 of 1% of the overhead (0.0075) as opposed to not doing so, and that in the typical case the overhead is less than 1/4 of 1% (0.0025). Therefore, for all practical purposes the overhead of applying this method is negligible. However, the benefits of doing so are great and provide the capability of true transaction order that are not found in prior art.

SELECTING THE DATA IN TRANSACTION ORDER

[48] In the embodiment of the present invention described herein above, both the change data and the commit SCN for each and every transaction that was captured have been recorded in the change tables 211, 213, 221, 223 and transaction table ~~240~~ 250, respectively. To obtain the contents of the change table 223 in the order in which the operation originally occurred, a database join operation between the change table 223 and the transaction ~~240~~ table 250 can be

used. For example, one implementation of the present invention may use the following SQL join operation, in which change table 223 has source table columns 231 named C1, C2, and C3:

```
SELECT TT.CSCN CSCN, CT1.RSID, CT1.C1 C1, CT1.C2 C2, CT1.C3 C3
FROM TT, CT1
WHERE TT.XID = CT1.XID
ORDER BY CSCN
```

[49] This statement relies on the fact that all SQL statements within a given transaction have the same value for a transaction identifier and that this value was stored in the transaction table ~~240~~ 250 in the same row as the commit system change number 243. Therefore a join across the two tables matches up all SQL statements belonging to a particular transaction with their associated commit system change number 243. Moreover, the SQL ORDER BY clause returns the change rows in increasing sorted order, according to their commit system change number, which is to say, in the original order in which the transactions committed.

On pages 19-20, please replace paragraphs 58-60 as follows:

[58] The network link 419 typically provides data communication through one or more networks to other data devices. For example, the network link 419 may provide a connection through local network 421 to a host computer 423, which has connectivity to a network 425 (e.g. a wide area network (WAN) or the global packet data communication network now commonly referred to as the "Internet") or to data equipment operated by a service provider. The local network 421 and network 425 both use electrical, electromagnetic, or optical signals to convey information and instructions. The signals through the various networks and the signals on network link 419 and through communication interface 417, which communicate digital data with computer system 400, are exemplary forms of carrier waves bearing the information and instructions.

[59] The computer system 400 can send messages and receive data, including program code, through the network(s), network link 419, and communication interface 417. In the Internet example, a server (not shown) might transmit requested code belonging to an application program for implementing an embodiment of the present invention through the network 425, local network 421 and communication interface 417. The processor ~~404~~ 403 may execute the transmitted code while being received and/or store the code in storage device ~~49~~ 409, or other

non-volatile storage for later execution. In this manner, computer system 400 may obtain application code in the form of a carrier wave.

[60] The term "computer-readable medium" as used herein refers to any medium that participates in providing instructions to the processor ~~404~~ 403 for execution. Such a medium may take many forms, including but not limited to non-volatile media, volatile media, and transmission media. Non-volatile media include, for example, optical or magnetic disks, such as storage device 409. Volatile media include dynamic memory, such as main memory 405. Transmission media include coaxial cables, copper wire, and fiber optics, including the wires that comprise bus 401. Transmission media can also take the form of acoustic, optical, or electromagnetic waves, such as those generated during radio frequency (RF) and infrared (IR) data communications. Common forms of computer-readable media include, for example, a floppy disk, a flexible disk, hard disk, magnetic tape, any other magnetic medium, a CD-ROM, CDRW, DVD, any other optical medium, punch cards, paper tape, optical mark sheets, any other physical medium with patterns of holes or other optically recognizable indicia, a RAM, a PROM, and EPROM, a FLASH-EPROM, any other memory chip or cartridge, a carrier wave, or any other medium from which a computer can read.

On page 20, please replace paragraph 62 as follows:

[62] Accordingly, embodiments of synchronous change data capture are described that ~~addresses~~ address the problems of maintaining strict and correct transaction ordering, which prior ~~solution~~ solutions have not been able to achieve. Furthermore, prior solutions also required post-processing or updates to many rows of the change table after the data was originally captured to record an approximate ordering of rows in the change table itself, an operation that was prohibitively expensive in prior solutions, and increased by an order of N, where N is the number of rows in the change table.